

**IN THE UNITED STATES DISTRICT COURT
FOR THE DISTRICT OF DELAWARE**

HONEYWELL INTERNATIONAL INC. and)
HONEYWELL INTELLECTUAL PROPERTIES INC.,)

Plaintiffs,)

v.)

C.A. No. 04-1338-KAJ

APPLE COMPUTER INC., et al,)

Defendants.)

**DECLARATION OF DIANA M. SANGALLI
IN SUPPORT OF ARIMA DISPLAY CORP.'S REPLY IN SUPPORT OF ITS
MOTION TO DISMISS FOR LACK OF PERSONAL JURISDICTION**

I, Diana M. Sangalli, declare as follows:

1. I am one of the attorneys representing Defendant Arima Display Corporation in the above-captioned lawsuit. I am submitting this declaration in support of Arima Display Corporation's Reply in Support of Its Motion to Dismiss for Lack of Personal Jurisdiction. I believe that the documents attached to this Declaration as Exhibits A and B are true and correct copies of what they purport to be.
2. Attached as Exhibit A is a true and correct copy of pages 1, 37 and 43 of a document that specifies the standard for the Hypertext Transfer Protocol referred to as HTTP/1.1. Section 10, which begins on page 37 of the document, of the standard sets forth Status Code definitions. Subsection 10.4.5, which is set forth on page 43 of the document, sets forth the definition of the HTTP 404 status code. Specifically, the HTTP 404 status code indicates that the server has not found anything that matches the requested Uniform Resource Indicator ("URI"). This document

can be downloaded from the World Wide Web Consortium ("W3C") web site found at <http://www.w3.org>.

3. Attached as Exhibit B is a true and correct copy of pages from a web site located at <http://msdn.microsoft.com>. These pages list the HTTP Status Codes and their definitions. Page 2 of this attachment shows that Status Code 404 represents HTTP_STATUS_NOT_FOUND which indicates that the server has not found anything that matches the requested URI.

4. I declare under penalty of perjury under the laws of the United States of America that the foregoing is true and correct.

Executed on April 17, 2006

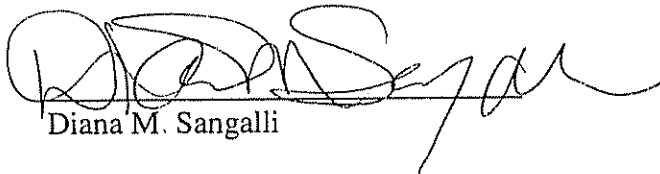

Diana M. Sangalli

EXHIBIT A

Network Working Group
Request for Comments: 2616
Obsoletes: 2068
Category: Standards Track

R. Fielding
UC Irvine
J. Gettys
Compaq/W3C
J. C. Mogul
Compaq
H. Frystyk
W3C/MIT
L. Masinter
Xerox
P. Leach
Microsoft
T. Berners-Lee
W3C/MIT
June, 1999

Hypertext Transfer Protocol -- HTTP/1.1

Status of this Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Copyright Notice

Copyright (C) The Internet Society (1999). All Rights Reserved.

Abstract

The Hypertext Transfer Protocol (HTTP) is an application-level protocol for distributed, collaborative, hypermedia information systems. It is a generic, stateless, protocol which can be used for many tasks beyond its use for hypertext, such as name servers and distributed object management systems, through extension of its request methods, error codes and headers [47]. A feature of HTTP is the typing and negotiation of data representation, allowing systems to be built independently of the data being transferred.

HTTP has been in use by the World-Wide Web global information initiative since 1990. This specification defines the protocol referred to as "HTTP/1.1", and is an update to RFC 2068 [33].

A successful response **SHOULD** be 200 (OK) if the response includes an entity describing the status, 202 (Accepted) if the action has not yet been enacted, or 204 (No Content) if the action has been enacted but the response does not include an entity.

If the request passes through a cache and the `Request-URI` identifies one or more currently cached entities, those entries **SHOULD** be treated as stale. Responses to this method are not cacheable.

9.8 TRACE

The TRACE method is used to invoke a remote, application-layer loop-back of the request message. The final recipient of the request **SHOULD** reflect the message received back to the client as the entity-body of a 200 (OK) response. The final recipient is either the origin server or the first proxy or gateway to receive a `Max-Forwards` value of zero (0) in the request (see section 14.31). A TRACE request **MUST NOT** include an entity.

TRACE allows the client to see what is being received at the other end of the request chain and use that data for testing or diagnostic information. The value of the `Via` header field (section 14.45) is of particular interest, since it acts as a trace of the request chain. Use of the `Max-Forwards` header field allows the client to limit the length of the request chain, which is useful for testing a chain of proxies forwarding messages in an infinite loop.

If the request is valid, the response **SHOULD** contain the entire request message in the entity-body, with a `Content-Type` of "message/http". Responses to this method **MUST NOT** be cached.

9.9 CONNECT

This specification reserves the method name `CONNECT` for use with a proxy that can dynamically switch to being a tunnel (e.g. SSL tunneling [44]).

10 Status Code Definitions

Each `Status-Code` is described below, including a description of which method(s) it can follow and any meta-information required in the response.

10.1 Informational 1xx

This class of status code indicates a provisional response, consisting only of the `Status-Line` and optional headers, and is terminated by an empty line. There are no required headers for this class of status code. Since HTTP/1.0 did not define any 1xx status codes, servers **MUST NOT** send a 1xx response to an HTTP/1.0 client except under experimental conditions.

A client **MUST** be prepared to accept one or more 1xx status responses prior to a regular response, even if the client does not expect a 100 (Continue) status message. Unexpected 1xx status responses **MAY** be ignored by a user agent.

Proxies **MUST** forward 1xx responses, unless the connection between the proxy and its client has been closed, or unless the proxy itself requested the generation of the 1xx response. (For example, if a proxy adds a "Expect: 100-continue" field when it forwards a request, then it need not forward the corresponding 100 (Continue) response(s).)

10.1.1 100 Continue

The client **SHOULD** continue with its request. This interim response is used to inform the client that the initial part of the request has been received and has not yet been rejected by the server. The client **SHOULD** continue by sending the remainder of the request or, if the request has already been completed, ignore this response. The server **MUST** send a final response after the request has been completed. See section 8.2.3 for detailed discussion of the use and handling of this status code.

10.4.5 404 Not Found

The server has not found anything matching the Request-URI. No indication is given of whether the condition is temporary or permanent. The 410 (Gone) status code SHOULD be used if the server knows, through some internally configurable mechanism, that an old resource is permanently unavailable and has no forwarding address. This status code is commonly used when the server does not wish to reveal exactly why the request has been refused, or when no other response is applicable.

10.4.6 405 Method Not Allowed

The method specified in the Request-Line is not allowed for the resource identified by the Request-URI. The response MUST include an Allow header containing a list of valid methods for the requested resource.

10.4.7 406 Not Acceptable

The resource identified by the request is only capable of generating response entities which have content characteristics not acceptable according to the accept headers sent in the request.

Unless it was a HEAD request, the response SHOULD include an entity containing a list of available entity characteristics and location(s) from which the user or user agent can choose the one most appropriate. The entity format is specified by the media type given in the Content-Type header field. Depending upon the format and the capabilities of the user agent, selection of the most appropriate choice MAY be performed automatically. However, this specification does not define any standard for such automatic selection.

Note: HTTP/1.1 servers are allowed to return responses which are not acceptable according to the accept headers sent in the request. In some cases, this may even be preferable to sending a 406 response. User agents are encouraged to inspect the headers of an incoming response to determine if it is acceptable.

If the response could be unacceptable, a user agent SHOULD temporarily stop receipt of more data and query the user for a decision on further actions.

10.4.8 407 Proxy Authentication Required

This code is similar to 401 (Unauthorized), but indicates that the client must first authenticate itself with the proxy. The proxy MUST return a Proxy-Authenticate header field (section 14.33) containing a challenge applicable to the proxy for the requested resource. The client MAY repeat the request with a suitable Proxy-Authorization header field (section 14.34). HTTP access authentication is explained in "HTTP Authentication: Basic and Digest Access Authentication" [43].

10.4.9 408 Request Timeout

The client did not produce a request within the time that the server was prepared to wait. The client MAY repeat the request without modifications at any later time.

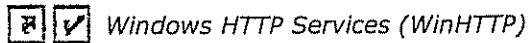
10.4.10 409 Conflict

The request could not be completed due to a conflict with the current state of the resource. This code is only allowed in situations where it is expected that the user might be able to resolve the conflict and resubmit the request. The response body SHOULD include enough information for the user to recognize the source of the conflict. Ideally, the response entity would include enough information for the user or user agent to fix the problem; however, that might not be possible and is not required.

Conflicts are most likely to occur in response to a PUT request. For example, if versioning were being used and the entity being PUT included changes to a resource which conflict with those made by an earlier (third-party) request, the server might use the 409 response to indicate that it can't complete the request. In this case, the response entity would likely contain a list of the differences between the two versions in a format defined by the response Content-Type.

EXHIBIT B

[MSDN Home](#) > [MSDN Library](#) > [Win32 and COM Development](#) > [Network Protocols](#) > [Windows HTTP Services \(WinHTTP\)](#) > [Windows HTTP Services \(WinHTTP\)](#) > [WinHTTP Reference](#) > [WinHTTP C/C++ API Reference](#) > [WinHTTP Constants](#)



HTTP Status Codes

These constants and corresponding values indicate HTTP status codes returned by servers on the Internet.

HTTP_STATUS_CONTINUE

100

The request can be continued.

HTTP_STATUS_SWITCH_PROTOCOLS

101

The server has switched protocols in an upgrade header.

HTTP_STATUS_OK

200

The request completed successfully.

HTTP_STATUS_CREATED

201

The request has been fulfilled and resulted in the creation of a new resource.

HTTP_STATUS_ACCEPTED

202

The request has been accepted for processing, but the processing has not been completed.

HTTP_STATUS_PARTIAL

203

The returned meta information in the entity-header is not the definitive set available from the originating server.

HTTP_STATUS_NO_CONTENT

204

The server has fulfilled the request, but there is no new information to send back.

HTTP_STATUS_RESET_CONTENT

205

The request has been completed, and the client program should reset the document view that caused the request to be sent to allow the user to easily initiate another input action.

HTTP_STATUS_PARTIAL_CONTENT

206

The server has fulfilled the partial GET request for the resource.

HTTP_STATUS_WEBDAV_MULTI_STATUS

207

During a World Wide Web Distributed Authoring and Versioning (WebDAV) operation, this indicates multiple status codes for a single response. The response body contains Extensible Markup Language (XML) that describes the status codes. For more information, see [HTTP Extensions for Distributed Authoring](#).

HTTP_STATUS_AMBIGUOUS

300

The requested resource is available at one or more locations.

HTTP_STATUS_MOVED

301

The requested resource has been assigned to a new permanent Uniform Resource Identifier (URI), and any future references to this resource should be done using one of the returned URIs.

HTTP_STATUS_REDIRECT

302

The requested resource resides temporarily under a different URI.

HTTP_STATUS_REDIRECT_METHOD

303

The response to the request can be found under a different URI and should be retrieved using a GET *HTTP verb* on that resource.

HTTP_STATUS_NOT_MODIFIED

304

The requested resource has not been modified.

HTTP_STATUS_USE_PROXY

305

The requested resource must be accessed through the proxy given by the location field.

HTTP_STATUS_REDIRECT_KEEP_VERB

307

The redirected request keeps the same HTTP verb. HTTP/1.1 behavior.

HTTP_STATUS_BAD_REQUEST

400

The request could not be processed by the server due to invalid syntax.

HTTP_STATUS_DENIED

401

The requested resource requires user authentication.

HTTP_STATUS_PAYMENT_REQ

402

Not implemented in the HTTP protocol.

HTTP_STATUS_FORBIDDEN

403

The server understood the request, but cannot fulfill it.

HTTP_STATUS_NOT_FOUND

404

The server has not found anything that matches the requested URI.

HTTP_STATUS_BAD_METHOD

405

The HTTP verb used is not allowed.

HTTP_STATUS_NONE_ACCEPTABLE

406

No responses acceptable to the client were found.

HTTP_STATUS_PROXY_AUTH_REQ

407

Proxy authentication required.

HTTP_STATUS_REQUEST_TIMEOUT

408

The server timed out waiting for the request.

HTTP_STATUS_CONFLICT

409

The request could not be completed due to a conflict with the current state of the resource. The user should resubmit with more information.

HTTP_STATUS_GONE

410

The requested resource is no longer available at the server, and no forwarding address is known.

HTTP_STATUS_LENGTH_REQUIRED

411

The server cannot accept the request without a defined content length.

HTTP_STATUS_PRECOND_FAILED

412

The precondition given in one or more of the request header fields evaluated to false when it was tested on the server.

HTTP_STATUS_REQUEST_TOO_LARGE

413

The server cannot process the request because the request entity is larger than the server is able to process.

HTTP_STATUS_URI_TOO_LONG

414

The server cannot service the request because the request URI is longer than the server can interpret.

HTTP_STATUS_UNSUPPORTED_MEDIA

415

The server cannot service the request because the entity of the request is in a format not supported by the requested resource for the requested method.

HTTP_STATUS_RETRY_WITH

449

The request should be retried after doing the appropriate action.

HTTP_STATUS_SERVER_ERROR

500

The server encountered an unexpected condition that prevented it from fulfilling the request.

HTTP_STATUS_NOT_SUPPORTED

501

The server does not support the functionality required to fulfill the request.

HTTP_STATUS_BAD_GATEWAY

502

The server, while acting as a gateway or proxy, received an invalid response from the upstream server it accessed in attempting to fulfill the request.

HTTP_STATUS_SERVICE_UNAVAIL

503

The service is temporarily overloaded.

HTTP_STATUS_GATEWAY_TIMEOUT

504

The request was timed out waiting for a gateway.

HTTP_STATUS_VERSION_NOT_SUP

505

The server does not support the HTTP protocol version that was used in the request message.

Requirements

Client	Requires Windows Vista, Windows XP SP1, or Windows 2000 Professional SP3 and later.
Server	Requires Windows Server "Longhorn", Windows Server 2003, or Windows 2000 Server SP3 and later.
Header	Declared in Winhttp.h.

See Also

[WinHTTP Versions](#)

[Send comments about this topic](#)

[Manage Your Profile](#) | [Legal](#) | [Contact Us](#) | [MSDN Flash Newsletter](#)

© 2006 Microsoft Corporation. All rights reserved. [Terms of Use](#) | [Trademarks](#) | [Privacy Statement](#)



UNITED STATES DISTRICT COURT
DISTRICT OF DELAWARE

CERTIFICATE OF SERVICE

I HEREBY CERTIFY that on April 17, 2006, I electronically filed the foregoing document with the Clerk of Court using CM/ECF which will send notification of such filing to the following, and the document is available for viewing and downloading from CM/ECF, and the document has also been served as indicated:

BY HAND

Thomas C. Grimm, Esquire
Leslie A. Polizoti, Esquire
Morris, Nichols, Arsht & Tunnell
Chase Manhattan Centre
1201 North Market Street
P.O. Box 1347
Wilmington, DE 19899
tgrimm@mnat.com
lpolizoti@mnat.com

Thomas L. Halkowski, Esquire
Fish & Richardson, P.C.
919 N. Market Street, Suite 1100
P.O. Box 1114
Wilmington, DE 19899-1114
halkowski@fr.com

Richard L. Horwitz, Esquire
Potter Anderson & Corroon
6th Floor, Hercules Plaza
1313 N. Market Street
P.O. Box 951
Wilmington, DE 19899
rhhorowitz@potteranderson.com

Adam Wyatt Poff, Esquire
Young, Conaway, Stargatt & Taylor
17th Floor, The Brandywine Building
1000 West Street
P.O. Box 391
Wilmington, DE 19899
apoff@ycst.com

Francis DiGiovanni, Esquire
Connolly, Bove, Lodge & Hutz LLP
The Nemours Building
1007 North Orange Street
P.O. Box 2207
Wilmington, DE 19899
fdigiovanni@cblh.com

Arthur G. Connolly, III, Esquire
Connolly Bove Lodge & Hutz
8th Floor, The Nemours Building
1007 N. Orange Street
P.O. Box 2207
Wilmington, DE 19899
aconnollyIII@cblh.com

David E. Moore, Esquire
Potter Anderson & Corroon
6th Floor, Hercules Plaza
1313 N. Market Street
P.O. Box 951
Wilmington, DE 19899
dmoore@potteranderson.com

Amy Evans, Esquire
Cross & Simons, LLC
Suite 1001, 913 N. Market Street
P.O. Box 1380
Wilmington, DE 19899
aevans@crosslaw.com

Frederick L. Cottrell, III, Esquire
Richards, Layton & Finger
One Rodney Square
P.O. Box 551
Wilmington, DE 19899
cottrell@rlf.com

Robert J. Katzenstein, Esquire
Smith, Katzenstein & Furlow LLP
800 Delaware Avenue, 7th Floor
P.O. Box 410
Wilmington, DE 19899
rkattenstein@skdelaware.com

Philip A. Rovner, Esquire
Potter Anderson & Corroon
6th Floor, Hercules Plaza
1313 N. Market Street
P.O. Box 951
Wilmington, DE 19899
provner@potteranderson.com


David J. Margules, Esquire
Bouchard, Margules, Friedlander
222 Delaware Avenue, Suite 1400
Wilmington, DE 19801
dmargules@BMF-law.com

John W. Shaw, Esquire
Young, Conaway, Stargatt & Taylor
17th Floor, The Brandywine Building
1000 West Street
P.O. Box 391
Wilmington, DE 19899
jshaw@ycst.com

Karen L. Pascale, Esquire
Young, Conaway, Stargatt & Taylor
17th Floor, The Brandywine Building
1000 West Street
Wilmington, DE 19899
kpascale@ycst.com

William J. Marsden, Jr., Esquire
Fish & Richardson, P.C.
919 N. Market Street, Suite 1100
P.O. Box 1114
Wilmington, DE 19899
marsden@fr.com

Monte Terrell Squire, Esquire
Young, Conaway, Stargatt & Taylor
17th Floor, The Brandywine Building
1000 West Street
P.O. Box 391
Wilmington, DE 19899
msquire@ycst.com


William J. Wade (#704)

Dated: April 17, 2006